CERTIFICATE OF TRANSMISSION

I hereby certify that this correspondence is being transmitted by either submission using the EFS WEB submission system, fax to the U.S. Patent and Trademark office to fax number 571-273-8300, or is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 to on February 18, 2010.

/Brian C. Kunzler/
Attorney for Applicants

PATENT
Docket No. TUC920030135US1

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants:	Stephen La Roux Blinick et al.	)
Serial No.:	10/717,941	)
Filed:	November 20, 2003	) Group Art Unit: 2192
For:	**APPARATUS, SYSTEM, AND METHOD FOR UPDATING AN EMBEDDED CODE IMAGE**	)
Examiner:	Ben C. Wang	)

EXAMINER'S AMENDMENT

Dear Examiner:

I approve these amendments as we discussed.

AMENDMENTS TO THE CLAIMS

Please amend the claims as indicated:

1. (Previously Presented) An apparatus for updating an embedded code image in a host bus adapter performing high speed data transfer between a host system and a storage system, comprising a processor, a main memory, and a temporary memory, the main memory and the temporary memory storing Microcode Reconstruct and Boot (MRB) format code images, and the apparatus further comprising:

the processor executing executable code stored on the main memory occupied by and used by an old code image comprising a first MRB format header, the executable code comprising:

a loader stored in the main memory and loading a new code image comprising a second MRB format header into the temporary memory;

a branch module stored in the main memory causing the processor to execute a bootstrap module within the new code image;

the bootstrap module accessing an old code image version number pointed to by the first MRB format header and a new code image version number pointed to by the second MRB format header;

the bootstrap module identifying:

incompatibilities between the old code image and the new code image from the old code image version number and the new code image version number;

a difference in initialization requirements for storage registers, memory, and hardware devices; and

a difference in size and location between the old code image and the new code image;

the bootstrap module further:

accessing capability information for the old code image and capability information for the new code image;

identifying a difference between the capability information; and

reconciling the incompatibilities by

changing an initialization order;

converting a format of a data structure of the old code image to a format compatible with a data structure of the new code image; and

associating persistent data of the old code image with the new code image, such that the persistent data is available in response to execution of a run-time segment of the new code image; and

a copy module copying the new code image into the main memory space occupied by the old code image.

2. (Previously Presented) The apparatus of claim 1, wherein the old code image is updated concurrently with normal execution of transactions by the apparatus.

3. (Previously Presented) The apparatus of claim 1, the executable code further comprising an initialization module initiating execution of the run-time segment.

4. (Canceled)

5. (Canceled)

6. (Canceled)

7. (Canceled)

8. (Canceled)

9. (Previously Presented) The apparatus of claim 31, wherein identifying incompatibilities further comprises identifying a difference between the format of the data structure used by the old code image and the format compatible with the data structure used by the new code image.

10. (Currently Amended) An apparatus for updating an embedded code image in a host bus adapter performing high speed data transfer between a host system and a storage system, comprising a processor, a main memory, and a temporary memory, the main memory and the temporary memory storing Microcode Reconstruct and Boot (MRB)~~MRB~~ format code images, the apparatus further comprising:

the processor executing executable code stored on the main memory occupied by and used by an old code image comprising a first MRB format header, the executable

code comprising

a loader loading a new code image comprising a second MRB format header into the temporary memory;

a branch module stored in the main memory causing the processor to execute a bootstrap module within the new code image;

the bootstrap module accessing an old code image version number pointed to by the first MRB format header and a new code image version number pointed to by the second MRB format header;

the bootstrap module identifying:

incompatibilities between the old code image and the new code image from the old code image version number and the new code image version number;

a difference in initialization requirements for storage registers, memory, and hardware devices; and

a difference in size and location between the old code image and the new code image;

the bootstrap module further:

accessing capability information for the old code image and capability information for the new code image;

identifying a difference between the capability information; and

reconciling the incompatibilities by

changing an initialization order;

converting a data structure of the old code image to a

format compatible with a data structure of the new code image prior to copying the new code image into the memory space occupied by the old code image; ~~and~~

associating persistent data of the old code image with the new code image, such that the persistent data is available in response to execution of a run-time segment of the new code image;~~.~~; and

copying the new code image into the main memory space occupied by the old code image.

11. (Previously Presented) The apparatus of claim 10, the executable code further comprising a copy module copying the new code image over the old code image in the main memory after the incompatibilities have been reconciled.

12. (Previously Presented) The apparatus of claim 10, wherein identifying incompatibilities further comprises identifying a difference between the format of the data structure used by the old code image and the format compatible with the data structure used by the new code image.

13. (Currently Amended) A system for updating an embedded code image in a host bus adapter performing high speed data transfer between a host system and a storage system , the system comprising a processor, a main memory, and a temporary memory, the main memory and

the temporary memory storing Microcode Reconstruct and Boot (MRB)~~MRB~~ format code images, the system further comprising:

the main memory storing an old code image comprising a first MRB format header;

the temporary memory separate from the main memory and storing a new code image comprising a second MRB format header;

a processor executing executable code of the old code image and the new code image, the executable code comprising;

a loader stored in the main memory and loading the new code image into the temporary memory;

a branch module stored in the main memory causing the processor to execute a bootstrap module within the new code image; and

the bootstrap module accessing an old code image version number pointed to by the first MRB format header and a new code image version number pointed to by the second MRB format header, the bootstrap module identifying:

incompatibilities between the old code image and the new code image from the old code image version number and the new code image version number;

a difference in initialization requirements for storage registers, memory, and hardware devices; and

a difference in size and location between the old code image and the new code image;

the bootstrap module further

accessing capability information for the old code image and capability information for the new code image;

identifying a difference between the capability information; and

reconciling the incompatibilities by

changing an initialization order;

converting a format of a data structure of the old code image to a format compatible with a data structure of the new code image; ~~and~~

associating persistent data of the old code image with the new code image, such that the persistent data is available in response to execution of a run-time segment of the new code image.; and

copying the new code image into the main memory space occupied by the old code image.

14. (Previously Presented) The system of claim 13, the executable code further comprising a copy module stored within the new code image overlaying the new code image in main memory with the old code image in response to reconciling the incompatibilities.

15. (Previously Presented) The system of claim 14, the loader loading the new code image into the temporary memory in response to an interrupt.

16. (Original) The system of claim 15, wherein the update module stores the old code image pointer and the new code image pointer in the data structure.

17. (Canceled)

18. (Previously Presented) The system of claim 33, wherein the bootstrap module further reconciles the incompatibilities by updating modules that interface with the new code image.

19. (Canceled)

20. (Currently Amended) A method for updating an embedded code image in a host bus adapter performing high speed data transfer between a host system and a storage system, comprising a processor, a main memory, and a temporary memory, the main memory and the temporary memory storing Microcode Reconstruct and Boot (MRB)~~MRB~~ format code images, the method further comprising:

loading, by use of the processor, a new code image comprising a second MRB format header into the temporary memory location separate from the main memory space occupied by and used by an old code image comprising a first MRB format header;

executing a bootstrap module within the new code image;

accessing an old code image version number pointed to by the first MRB format header and a new code image version number pointed to by the second MRB format header;

identifying, using the bootstrap module:

incompatibilities between the old code image and the new code image from the old code image version number and the new code image version number:

a difference in initialization requirements for storage registers, memory, and hardware devices; and

a difference in size and location between the old code image and the new code image;

accessing, using the bootstrap module, capability information for the old code image and capability information for the new code image;

identifying, using the bootstrap module: a difference between the capability information;

reconciling, using the bootstrap module executed by the processor, the incompatibilities by

changing an initialization order;

converting a data structure of the old code image to a format compatible with a data structure of the new code image using bootstrap code of the new code image; and

associating persistent data of the old code image with the new code image, such that the persistent data is available in response to execution of a run-time segment of the new code image; and

copying the new code image into the main memory space occupied by the old code image.

21. (Previously Presented) The method of claim 20, wherein the new code image is copied to the main memory concurrently with execution of regular computer operations.

22. (Previously Presented) The method of claim 20, further comprising initiating execution of the run-time segment of the new code image.

23. (Previously Presented) The method of claim 20, wherein the new code image is copied into the main memory after the incompatibilities are reconciled.

24. (Canceled)

25. (Canceled)

26. (Canceled)

27. (Canceled)

28. (Previously Presented) The method of claim 34, wherein identifying incompatibilities further comprises identifying a difference between the format of the data structure used by the old code image and the format compatible with the data structure used by the new code image.

29. (Currently Amended) An apparatus for updating an embedded code image in a host bus adapter performing high speed data transfer between a host system and a storage system, the apparatus comprising a processor, a main memory, and a temporary memory, the main memory and the temporary memory storing Microcode Reconstruct and Boot (MRB)~~MRB~~ format code images and the apparatus further comprising:

the processor executing executable code stored on the main memory occupied by and used by an old code image comprising a first MRB format header, the executable code comprising:

means for loading a new code image comprising a second MRB format header into the temporary memory;

means for causing the processor to execute a bootstrap module within the new code image;

means within the bootstrap module for accessing an old code image version number pointed to by the first MRB format header and a new code image version number pointed to by the second MRB format header;

means within the bootstrap module for identifying:

incompatibilities between the old code image and the new code image from the old code image version number and the new code image version number;

a difference in initialization requirements for storage registers, memory, and hardware devices; and

a difference in size and location between the old code image and the new code image;

means with the bootstrap module for accessing capability information for the old code image and capability information for the new code image and identifying a difference between the capability information;

means within the bootstrap module for reconciling the incompatibilities by

changing an initialization order;

converting a format of a data structure of the old code image to a format

compatible with a data structure of the new code image using bootstrap code of the new code image; and

associating persistent data of the old code image with the new code image, such that the persistent data is available in response to execution of a run-time segment of the new code image; and

means for copying the new code image into the memory space occupied by the old code image.

30. (Currently Amended) An article of manufacture comprising a program storage medium readable by a processor and embodying one or more instructions executable by a processor to perform a method for updating an embedded code image in a host bus adapter performing high speed data transfer between a host system and a storage system by use of the processor, a main memory, and a temporary memory, the main memory and the temporary memory storing Microcode Reconstruct and Boot (MRB)~~MRB~~ format code images, the method comprising:

loading, by use of the processor, a new code image comprising a second MRB format header into the temporary memory location separate from a memory space occupied by and used by an old code image comprising a first MRB format header;

causing the processor to execute a bootstrap module within the new code image;

accessing, using the bootstrap module, an old code image version number pointed to by the first MRB format header and a new code image version number pointed to by the second MRB format header;

identifying, using the bootstrap module:

incompatibilities between the old code image and the new code image from the old code image version number and the new code image version number;

a difference in initialization requirements for storage registers, memory, and hardware devices; and

a difference in size and location between the old code image and the new code image;

accessing, by using the bootstrap module, capability information for the old code image and capability information for the new code image and identifying a difference between the capability information;

reconciling, using the bootstrap module executed by the processor, the incompatibilities by

changing an initialization order;

converting a format of a data structure of the old code image to a format compatible with a data structure of the new code image using bootstrap code of the new code image; and

associating persistent data of the old code image with the new code image, such that the persistent data is available in response to execution of a run-time segment of the new code image; and

copying the new code image into the main memory space occupied by the old code image.

31. (Currently Amended) The apparatus of claim 1, wherein the loader configures the temporary memory so that the executable code is executed directly from the temporary memory, the

executable code further comprising an update module stored in the main memory maintaining an old code image pointer, a new code image pointer, capability fields storing the capability information, the old code image version number, the new code image version number, the old code image pointer, the new code image pointer, the capability fields, the old code image version number, and the new code image version number used by the bootstrap module, wherein the bootstrap module follows the old code image pointer to locate an old code image header and a version field within the old code image header and follows the new code image pointer to locate a new code image header and a version field within the new code image header, the old code image header and the new code image header are organized according to the MRB format, the bootstrap module reading the capability information from the old code image and the new code image and storing the capability information of the old code image and the new code image in the capability fields, the capability information comprising an indication that an EMULEX FLASH Random Access Memory (RAM)~~RAM~~ is provided, the persistent data comprising login tables, and wherein reconciling incompatibilities between the old code image and the new code image further comprises adjusting configuration settings and parameter lists.

32. (Currently Amended) The apparatus of claim 10, wherein the loader configures the temporary memory so that the executable code is executed directly from the temporary memory, the executable code further comprising an update module stored in the main memory maintaining an old code image pointer, a new code image pointer, capability fields storing the capability information, the old code image version number, the new code image version number, the old code image pointer, the new code image pointer, the capability fields, the old code image version number, and the new code image version number used by the bootstrap module, wherein

the bootstrap module follows the old code image pointer to locate an old code image header and a version field within the old code image header and follows the new code image pointer to locate a new code image header and a version field within the new code image header, the old code image header and the new code image header are organized according to the MRB format, the bootstrap module reading the capability information from the old code image and the new code image and storing the capability information of the old code image and the new code image in the capability fields, the capability information comprising an indication that an EMULEX FLASH Random Access Memory (RAM)~~RAM~~ is provided, the persistent data comprising login tables, and wherein reconciling incompatibilities between the old code image and the new code image further comprises adjusting configuration settings and parameter lists and associating persistent data of the old code image with the new code image.

33. (Currently Amended) The system of claim 13, wherein the loader configures the temporary memory so that the executable code is executed directly from the temporary memory, the executable code further comprising an update module stored in the main memory maintaining an old code image pointer, a new code image pointer, capability fields storing the capability information, the old code image version number, the new code image version number, the old code image pointer, the new code image pointer, the capability fields, the old code image version number, and the new code image version number used by the bootstrap module, wherein the bootstrap module follows the old code image pointer to locate an old code image header and a version field within the old code image header and follows the new code image pointer to locate a new code image header and a version field within the new code image header, the old code image header and the new code image header are organized according to the MRB format, the

bootstrap module reading the capability information from the old code image and the new code image and storing the capability information of the old code image and the new code image in the capability fields, the capability information comprising an indication that an EMULEX FLASH Random Access Memory (RAM)~~RAM~~ is provided, the persistent data comprising login tables, and wherein reconciling incompatibilities between the old code image and the new code image further comprises adjusting configuration settings and parameter lists.

34. (Currently Amended) The method of claim 20, the method further comprising configuring the temporary memory so that the executable code is executed directly from the temporary memory, maintaining an old code image pointer, a new code image pointer, capability fields storing the capability information, the old code image version number, the new code image version number, the old code image pointer, the new code image pointer, the capability fields, the old code image version number, and the new code image version number used by the bootstrap module, wherein the bootstrap module follows the old code image pointer to locate an old code image header and a version field within the old code image header and follows the new code image pointer to locate a new code image header and a version field within the new code image header, the old code image header and the new code image header are organized according to the MRB format, the bootstrap module reading the capability information from the old code image and the new code image and storing the capability information of the old code image and the new code image in the capability fields, the capability information comprising an indication that an EMULEX FLASH Random Access Memory (RAM)~~RAM~~ is provided, the persistent data comprising login tables, wherein reconciling incompatibilities between the old

code image and the new code image further comprises adjusting configuration settings and parameter lists.

35. (Currently Amended) The article of manufacture of claim 30, the method further comprising configuring the temporary memory so that the executable code is executed directly from the temporary memory, maintaining an old code image pointer, a new code image pointer, capability fields storing the capability information, the old code image version number, the new code image version number, the old code image pointer, the new code image pointer, the capability fields, the old code image version number, and the new code image version number used by the bootstrap module, wherein the bootstrap module follows the old code image pointer to locate an old code image header and a version field within the old code image header and follows the new code image pointer to locate a new code image header and a version field within the new code image header, the old code image header and the new code image header are organized according to the MRB format, the bootstrap module reading the capability information from the old code image and the new code image and storing the capability information of the old code image and the new code image in the capability fields, the capability information comprising an indication that an EMULEX FLASH Random Access Memory (RAM)~~RAM~~ is provided, the persistent data comprising login tables, and wherein reconciling incompatibilities between the old code image and the new code image further comprises adjusting configuration settings and parameter lists and associating persistent data of the old code image with the new code image.

36. (Currently Amended) The apparatus of claim 29, wherein loading means configures the temporary memory so that the executable code is executed directly from the temporary memory,

the executable code further comprising maintaining means stored in the main memory maintaining an old code image pointer, a new code image pointer, capability fields storing the capability information, the old code image version number, the new code image version number, the old code image pointer, the new code image pointer, the capability fields, the old code image version number, and the new code image version number used by the bootstrap module, wherein the bootstrap module follows the old code image pointer to locate an old code image header and a version field within the old code image header and follows the new code image pointer to locate a new code image header and a version field within the new code image header, the old code image header and the new code image header are organized according to the MRB format, the bootstrap module reading the capability information from the old code image and the new code image and storing the capability information of the old code image and the new code image in the capability fields, the capability information comprising an indication that an EMULEX FLASH Random Access Memory (RAM)~~RAM~~ is provided, the persistent data comprising login tables, and wherein reconciling incompatibilities between the old code image and the new code image further comprises adjusting configuration settings and parameter lists.